National Aeronautics and Space Administration

HRP Investigators Workshop - 2017

IMPLICIT FORMULATION OF MUSCLE DYNAMICS IN OPENSIM

Brad Humphreys[1]
Chris Dembia[2]
Beth Lewandowski[3]
Antonie van den Bogert[4]

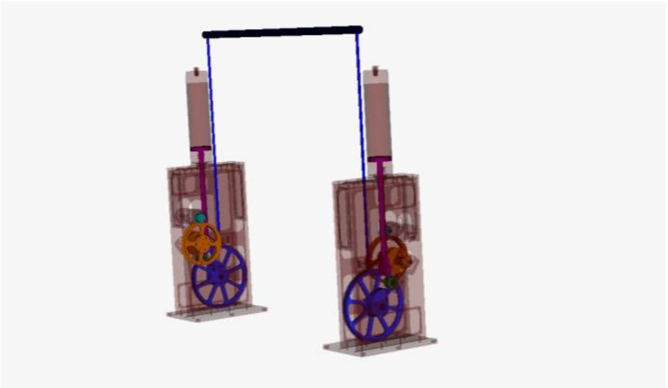
[1]ZIN Technologies
[2]Stanford University
[3]NASA GRC
[4]Cleveland State University

Why Predictive Modeling?

Provides a ***virtual subject*** for which we are able to perform studies such as sensitivity to device loading and vibration isolation ***without the need for laboratory kinematic or kinetic test data***.

1) Inform Device Design
    - Efficacy and Performance
        - Microgravity environment is unique
        - Exercise device loads typically also include replacing the body weight of the astronaut
    - Interface with Flight Vehicle
        - Operational Limitations
        - Structural Limitations
        - Vibration Isolation
2) Inform Other Models (Digital Astronaut Program)
    - Bone Adaption Models
    - Muscle Adaption Models

Previously Developed OpenSim Exercise Device Model



Direct Collocation

- DC In 3 slides as presented by Dr. van den Bogert

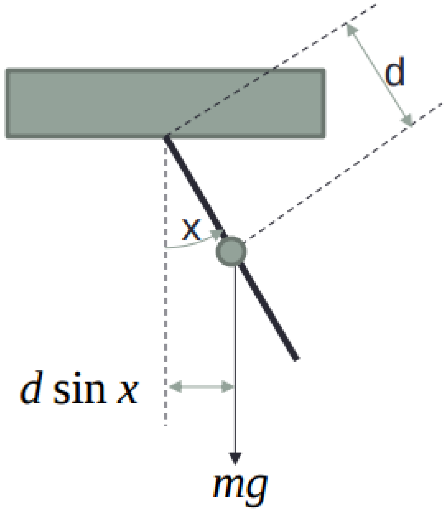
1-DOF pendulum

- Variables
  - angle x, torque u
- System dynamics:

$$I\,\ddot{x} = -mgd\sin x + u$$

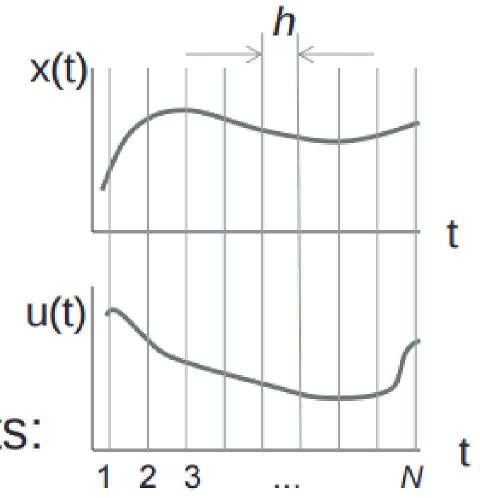
- Task constraints for swing-up problem:
  - Initial state $x(0) = 0, \quad \dot{x}(0) = 0$
  - Final state $x(T) = \pi, \quad \dot{x}(T) = 0$
- Cost function: integral of squared torque

$$\int_0^T u(t)^2\,dt$$



Acknowledgement: Slides originally created/presented by Dr. van den Bogert (CSU)

Temporal discretization



- Direct collocation: *N* nodes
- Time step $h = T/(N-1)$
- States $x_1, x_2 \ldots x_N$ controls $u_1, u_2 \ldots u_N$
- Dynamics becomes algebraic constraints:

$$I\,\ddot{x} = -mgd\sin x + u \Rightarrow I\,\frac{x_{i+1} - 2x_i + x_{i-1}}{h^2} = -mgd\sin x_i + u_i$$

- Cost function becomes an algebraic function:

$$\sum_i u_i^2$$

- Time step *h* usually much larger than in ODE solver
  - Convergence study to decide how many nodes are needed
  - For human gait cycle: N=50 or N=100 is typically good enough

Constrained optimization problem

- Unknowns

$$\mathbf{y} = (x_1, x_2 \ldots x_N, u_1, u_2 \ldots u_N)^T$$

- Minimize

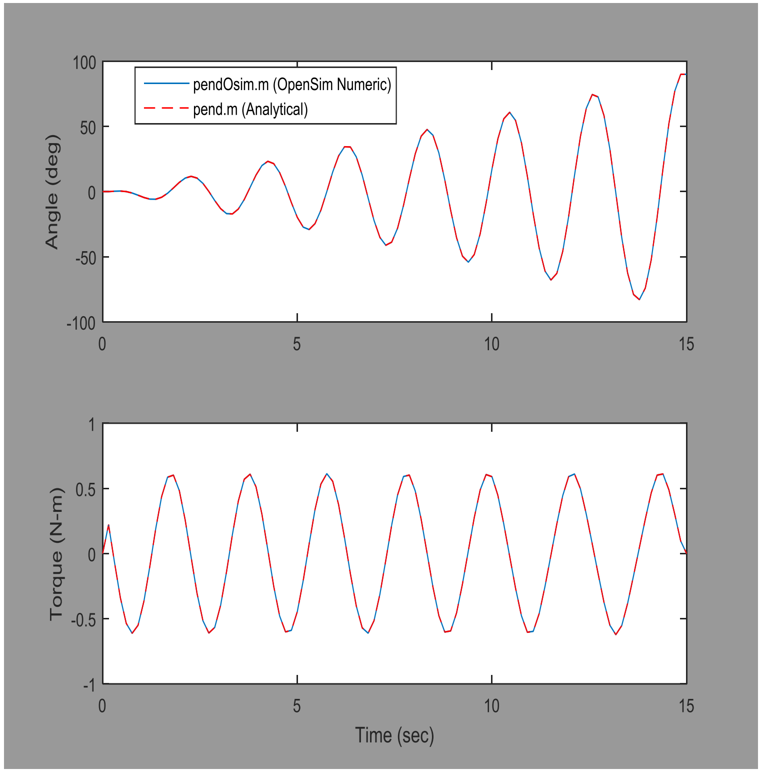
$f(\mathbf{y})$ cost function

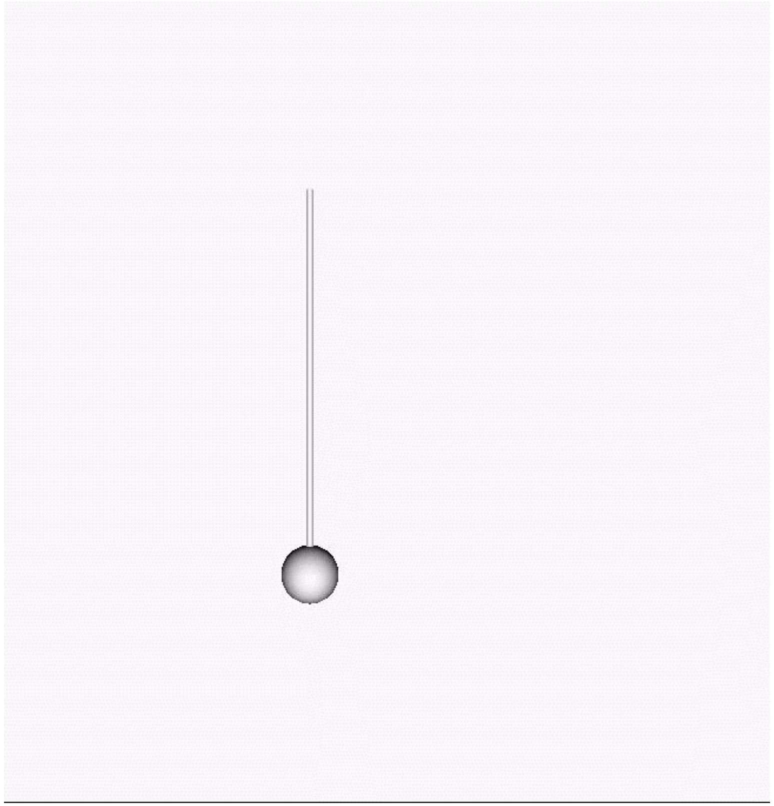
- Subject to

$\mathbf{c}(\mathbf{y}) = 0$ task constraints and dynamics constraints

Direct Collocation

- Pendulum implemented in OpenSim
  - Used to develop API code
  - Instead of dynamic equation of motion, model states come from OpenSim API
- OpenSim Results match MATLAB solution

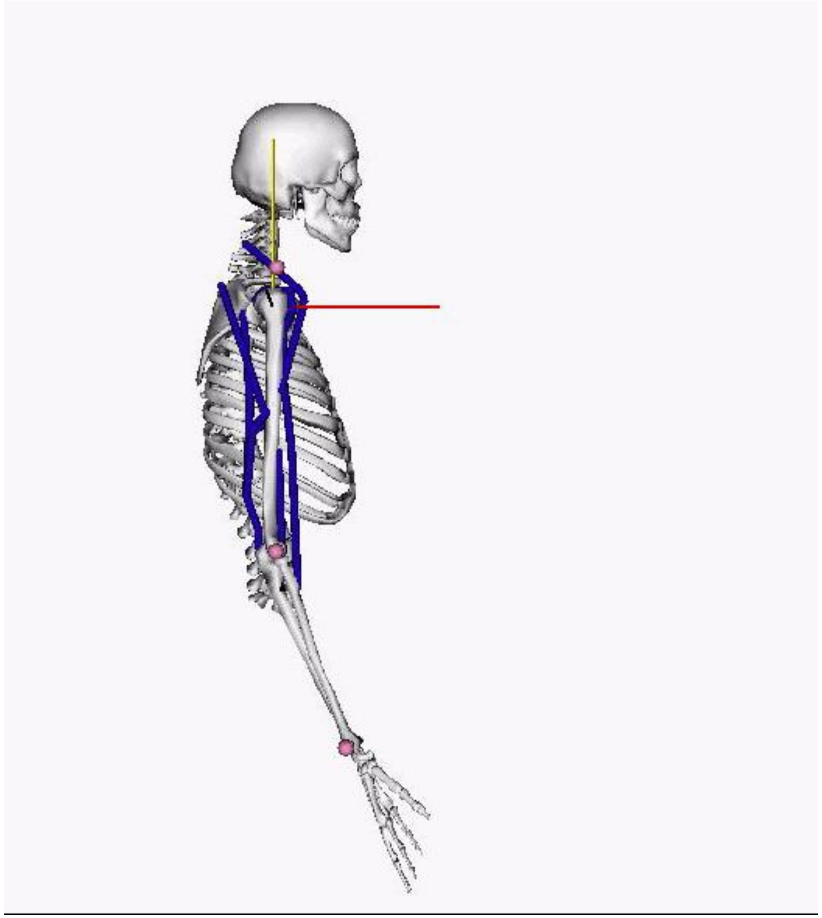




Simulation task was to raise to $\pi/2$

Direct Collocation

With Muscles…..



Direct Collocation

Direct Collocation is a way to formulate a problem.

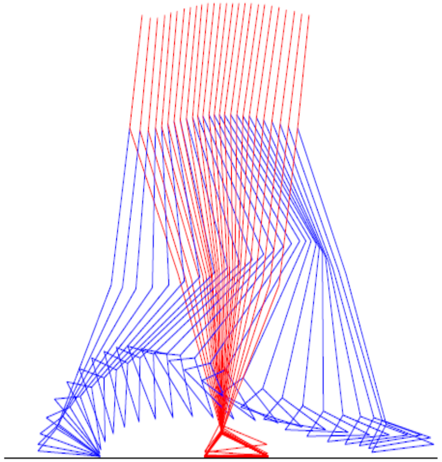
A solver is still needed to actually solve the problem.

- IPOPT, SNOPT, GPOPS, fmincon

These solvers utilize the Jacobian (partial derivatives) of:

- The objective function to each of the inputs
- The constraint equations to each of the inputs

The Jacobian is essentially the gradient around a trial solution.



Two ways to get the Jacobian

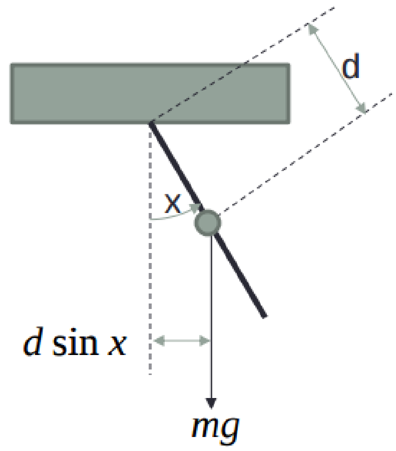
1) Take the derivative of the analytical functions (if they are available)
2) Use finite differences (computationally expensive)

Enough about Collocation for a bit ….. Let’s talk Implicit Dynamics.

Explicit Dynamics

Explicitly Solve

$$I\ddot{x} = -mgdsin(x) + u$$



Solve for acceleration (and then integrate):

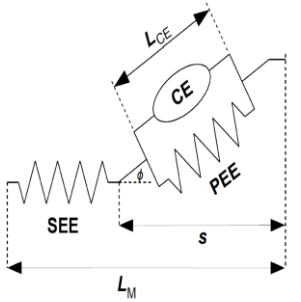
$$\ddot{x} = \frac{-mgdsin(x) + u}{I}$$

Problem: if inertia gets small, problem becomes “stiff”

Small changes in control *u* results in large accelerations and then small time steps are needed.

This also happens in muscles…..

Muscle Dynamics



Force Balance Equation for a Muscle

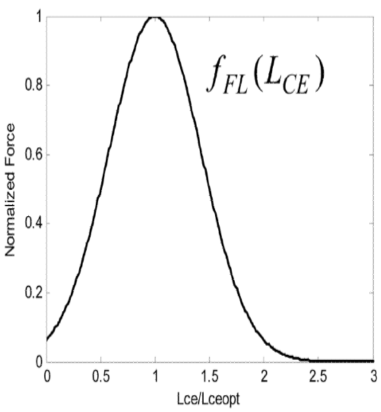
$$\underbrace{\left[a \cdot F_{max} \cdot f_{FL}(L_{CE}) \cdot f_{FV}(\dot{L_{CE}}) + f_{PEE}(L_{CE})\right]\cos\phi}_{\textit{Force In Fiber}} - \underbrace{f_{SEE}(L_M - L_{CE}\cos\phi)}_{\textit{Force In Tendon}} = 0$$

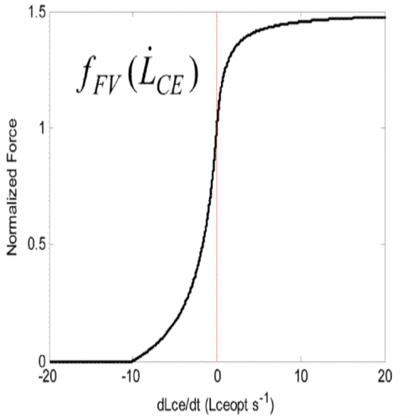
Solving to get Fiber Force-Velocity Relationship

$$f_{FV}(\dot{L_{CE}}) = \frac{f_{SEE}(L_M - L_{CE}\cos\phi)}{a \cdot F_{max} \cdot f_{FL}(L_{CE}) \cdot \cos\phi} - \frac{f_{PEE}(L_{CE})}{a \cdot F_{max} \cdot f_{FL}(L_{CE})} = f(L_M, L_{CE}, \phi, a)$$

Problem becomes stiff (singularities) when

- Activation, a ≈ 0
- The muscle gets long $f_{FL}(L_{CE}) \approx 0$
- Pennation angle, p ≈ 90deg





Implicit Dynamics

In Implicit, we don't invert, we root solve:

$$I\ddot{x} + mgdsin(x) - u = 0 \qquad\qquad F_{Res} = f(\ddot{x}, x, u) = f(\boldsymbol{y})$$

To root solve in Newton Raphson

1) Guess values for ***y*** *and calculate the residual.*

2) Calculate the Newton Step, δ and add it to the original guess to get the next guess.

$$y_{next\ guess} = y_{guess} + \delta = y_{guess} + \frac{f(\ddot{x}, x, u)}{\acute{f}(\ddot{x}, x, u)}$$

3) Loop until the δ gets small.

But wait a second…… What's is $\acute{f}(\ddot{x}, x, u)$?

- That's the derivative of the residual wrt to each of the inputs.

$$f(\ddot{x}, \acute{x}, u) = \frac{df}{d\boldsymbol{y}}$$

- That's the information we want for the Jacobian for the Direct Collocation Optimizer Solvers too!

Implicit Dynamics

So if we formulate muscle dynamics in an implicit form:

$$\left[a \cdot F_{max} \cdot f_{FL}(L_{CE}) \cdot f_{FV}\left(\dot{L_{CE}}\right) + f_{PEE}(L_{CE})\right]cos\phi - f_{SEE}(L_M - L_{CE}cos\phi) = F_{Res} = 0$$

$$F_{Res} = f(L_M, L_{CE}, \phi, a)$$

Not only could we use the Jacobian during the forward solving of the dynamics, the Jacobian elements become available for use in optimization solvers.

<u>Some added bonuses:</u>

1) We don't have to invert the force-length and force velocity relationships, we can use them directly.
2) We can have zero activation for muscles.
3) Implicit dynamics Jacobians have a more sparse structure.

<u>Cons</u>

1) Solvers can become stuck in minima or unable to solve.
2) Initial guess is important (we can work around this by starting with good known states and using previous states as guess).

An Implicit Muscle Model

1) Continuous Derivatives

2) State solution based approach using projected length of contractile element.

Procedia IUTAM. 2011 January 1; 2(2011): 297–316. doi:10.1016/j.piutam.2011.04.027.

Implicit methods for efficient musculoskeletal simulation and optimal control

Antonie J. van den Bogert[a,b,*], **Dimitra Blana**[b,c], and **Dieter Heinrich**[d]

[a]Orchard Kinetics LLC, Cleveland, OH, USA

[b]Department of Biomedical Engineering, Case Western Reserve University, Cleveland, OH, USA

[c]Department of Sport and Exercise Science, Aberystwyth University, Aberystwyth, Wales, UK

[d]Department of Sport Science, University of Innsbruck, Innsbruck, Austria

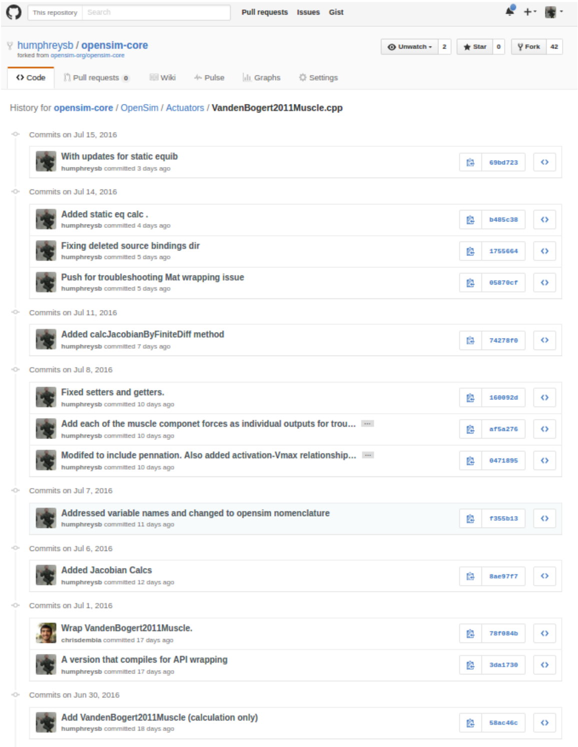
Abstract

The ordinary differential equations for musculoskeletal dynamics are often numerically stiff and highly nonlinear. Consequently, simulations require small time steps, and optimal control problems are slow to solve and have poor convergence. In this paper, we present an implicit formulation of musculoskeletal dynamics, which leads to new numerical methods for simulation

Project Status

Done To Date

- OpenSim forked and new muscle model created
- Test cases/models developed in MATLAB and coded into C++
  - Derivative Check (Muscle analytical derivs = finite element derivs?)
  - Forward Simulation of Tug-of-War
  - Direct Collocation Optimization
  - Energy Conservation
  - Doxygen Documentation Drafted
  - Figures generated via MATLAB API calls to Muscle Model
- Applied to Arm28 Model



To Do

- Apply to Squat Model
- Working other implicit capabilities with Chris Dembia at Stanford/OpenSim
  - Plan and code for generalized forces
  - Plan and code for constraints
  - **Long Term** Need a means to provide fast moment arms (possibly pre-process and create polynomials)